MDSC -102-FINAL LAB

FLIGHT PRICE DATASET

The price of an Airline Ticket is affected by a number of factors, such as flight duration, days left for departure, arrival time and departure time etc. Airline organisations may diminish the cost at the time they need to build the market and at the time when the tickets are less accessible. They may maximise the costs

The features of the dataset are
1.Airline
2 Flight-the flight type of each airline
3 Source city: the source city of each flight
4 Departure city: the departure city of the each flight
5 stops: the number of stops each flight has
6 arrival time: the arrival time of the each flight
7 Destination city: the destination city of the each flight journey
8 class: the class of the each flight weather it is business clas or the economy class
9: Duration: the duration of each flight journey how much time it takes
10:days_left:the days left for for the journey
11:price: the price of each flight route

The packages used

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import scipy as sp
import scipy.stats as stats
import statistics
```

The dataset

	Unnamed: 0	airline	flight	source_city	departure_time	stops	arrival_time	destination_city	class	duration	days_left	price
0	0	SpiceJet	SG-8709	Delhi	Evening	zero	Night	Mumbai	Economy	2.17	1	5953
1	1	SpiceJet	SG-8157	Delhi	Early_Morning	zero	Morning	Mumbai	Economy	2.33	1	5953
2	2	AirAsia	I5-764	Delhi	Early_Morning	zero	Early_Morning	Mumbai	Economy	2.17	1	5956
3	3	Vistara	UK-995	Delhi	Morning	zero	Afternoon	Mumbai	Economy	2.25	1	5955
4	4	Vistara	UK-963	Delhi	Morning	zero	Morning	Mumbai	Economy	2.33	1	5955
...	...	...	...	...	...	...	...	...	...	...	...	...
300148	300148	Vistara	UK-822	Chennai	Morning	one	Evening	Hyderabad	Business	10.08	49	69265
300149	300149	Vistara	UK-826	Chennai	Afternoon	one	Night	Hyderabad	Business	10.42	49	77105
300150	300150	Vistara	UK-832	Chennai	Early_Morning	one	Night	Hyderabad	Business	13.83	49	79099
300151	300151	Vistara	UK-828	Chennai	Early_Morning	one	Evening	Hyderabad	Business	10.00	49	81585
300152	300152	Vistara	UK-822	Chennai	Morning	one	Evening	Hyderabad	Business	10.08	49	81585

The shape of the dataset is

```
df.shape
✓ 0.0s
(300153, 12)
```

The preprocessing of the data

First we checked for the null values

```
print("Missing values before preprocessing:")
print(df.isnull().sum())

✓ 2.0s

Missing values before preprocessing:
Unnamed: 0          0
airline             0
flight              0
source_city         0
departure_time      0
stops               0
arrival_time        0
destination_city    0
class               0
duration            0
days_left           0
price               0
dtype: int64
```

This shows that we have no missing values
For example we filled the missing values for the duration features with the mean value for the duration feature

```
df['duration'].fillna(df['duration'].mean(), inplace=True)
df["duration"]
```

✓ 0.0s

```
0          2.17
1          2.33
2          2.17
3          2.25
4          2.33
          ...
300148    10.08
300149    10.42
300150    13.83
300151    10.00
300152    10.08
Name: duration, Length: 300153, dtype: float64
```

Next we dropped the duplicate values

```
# Check and handle duplicates
df.drop_duplicates(inplace=True)
```

✓ 1.5s

We get the statistics for the above dataset using the describe function

```
df.describe()
```

✓ 0.1s

	Unnamed: 0	duration	days_left	price
count	300153.000000	300153.000000	300153.000000	300153.000000
mean	150076.000000	12.221021	26.004751	20889.660523
std	86646.852011	7.191997	13.561004	22697.767366
min	0.000000	0.830000	1.000000	1105.000000
25%	75038.000000	6.830000	15.000000	4783.000000
50%	150076.000000	11.250000	26.000000	7425.000000
75%	225114.000000	16.170000	38.000000	42521.000000
max	300152.000000	49.830000	49.000000	123071.000000

The next we will tell the most popular airline by using a countplot

We get the information of the dataset features by using the info

```
df.info()
✓ 0.8s
```

```
class 'pandas.core.frame.DataFrame'>
angeIndex: 300153 entries, 0 to 300152
ata columns (total 12 columns):
#   Column            Non-Null Count   Dtype
--  ------            --------------   -----
0   Unnamed: 0        300153 non-null  int64
1   airline           300153 non-null  object
2   flight            300153 non-null  object
3   source_city       300153 non-null  object
4   departure_time    300153 non-null  object
5   stops             300153 non-null  object
6   arrival_time      300153 non-null  object
7   destination_city  300153 non-null  object
8   class             300153 non-null  object
9   duration          300153 non-null  float64
10  days_left         300153 non-null  int64
11  price             300153 non-null  int64
types: float64(1), int64(3), object(8)
emory usage: 27.5+ MB
```

Visualisations

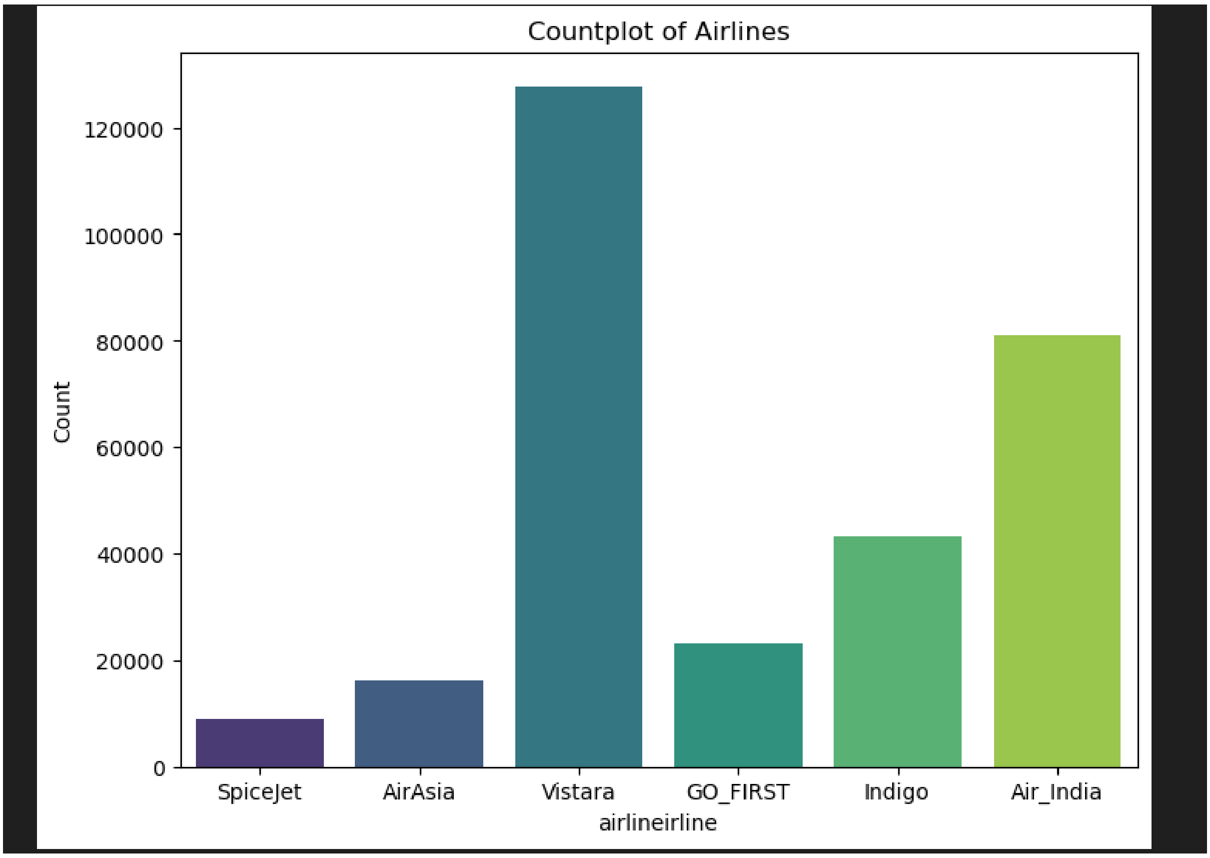
This will tell us the most number of airlines operating in the route

```
plt.figure(figsize=(8, 6))
sns.countplot(x='airline', data=df, palette='viridis')


plt.xlabel('airlineirline')
plt.ylabel('Count')
plt.title('Countplot of Airlines')


plt.show()
✓ 1.5s
```

The countplot for the following visualisation is



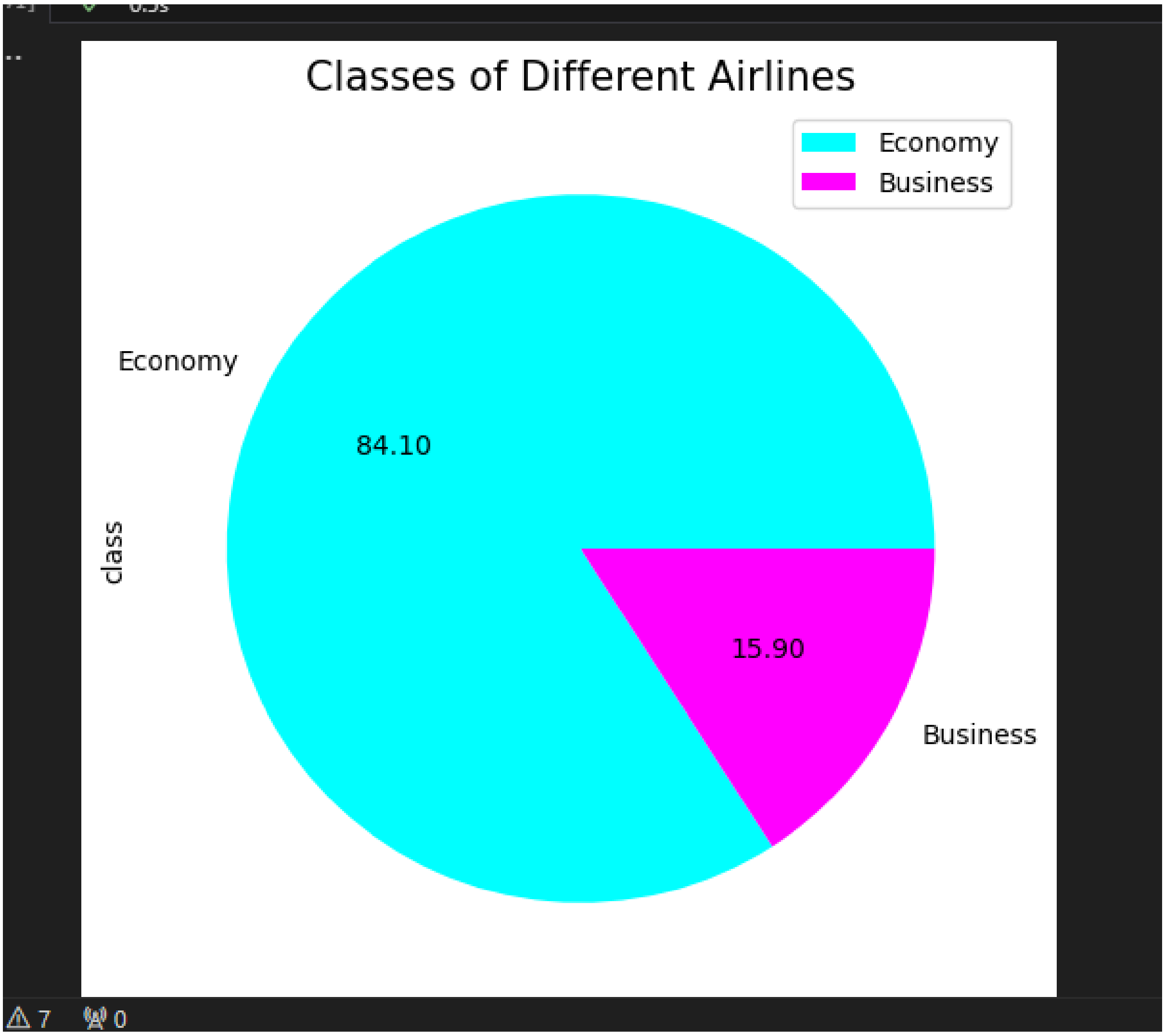
From the countplot we can infer that the airline vistara is having the most number of airline

The next we counted the number of class in the flights,we counted the number of economy class and the business class of flights

```
df2=df.groupby(['flight','airline','class'],as_index=False).count()
df2['class'].value_counts()
```

✓ 0.8s

```
Economy     1560
Business     295
Name: class, dtype: int64
```



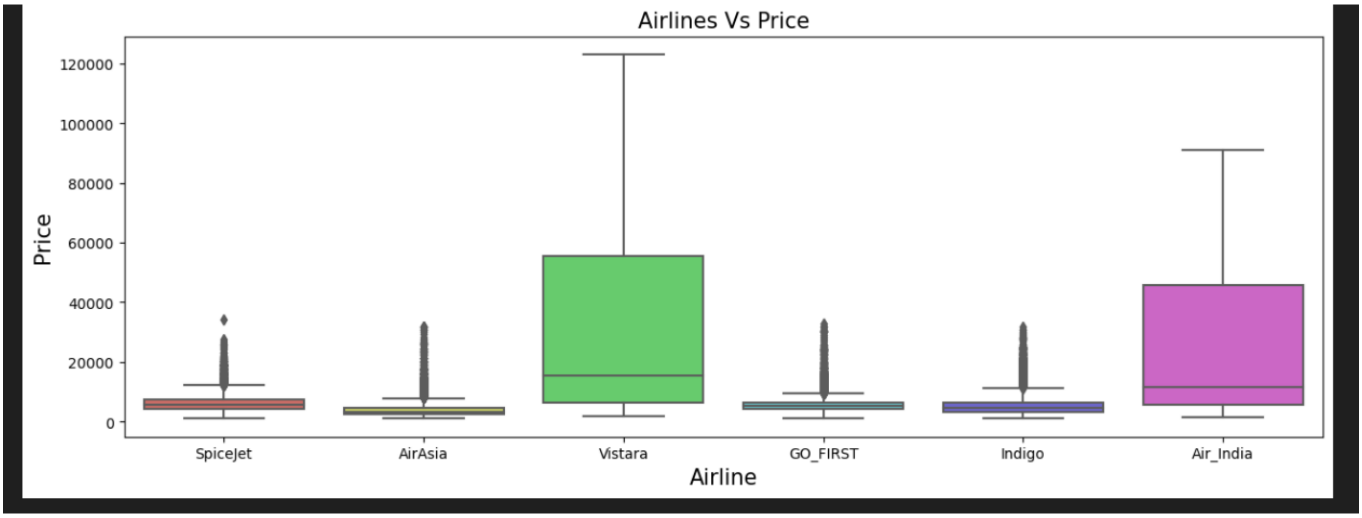
The next we plotted the percentage of business and the economy classes of the airlines and we came to a conclusion that most of the airlines are having the economy class
We used a pie chart for the visualisation

Box plot

Next we looked at how the price vary with each flights
We used a box plot to visualise how the price vary for each flights
We can infer the median- the line inside the box represents the median of the precise distribution

If the median is close to the bottom of the box it shows that the significant portion of the flight has the lower price

From the box plots we can understand the inter quartile range of the price distribution of the box and the we can also infer the outliers and skewness



From the above we can see that the airline vistara has the highest price
Spice jet,air asia,go first indigo,air india has the outliers
The price vary largely for vistara and air india

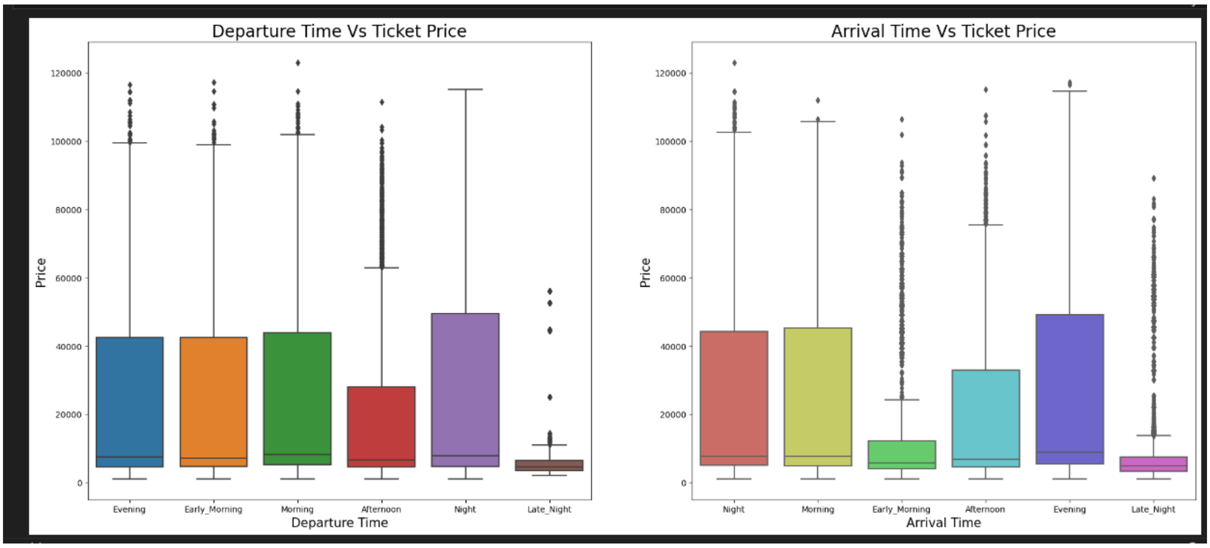
Box plot the distribution of departure time vs the price,arrival time vs the ticket price

Next we used the box plot for the distribution of the features departure time vs the ticket price and the arrival time and the ticket price
From this we can infer that the ticket price is high for the flights which are having the arrival time at night

The ticket price is same for the flights having the departure time as morning and early morning

The ticket price is more for the flight that are having the arrival time as the evening

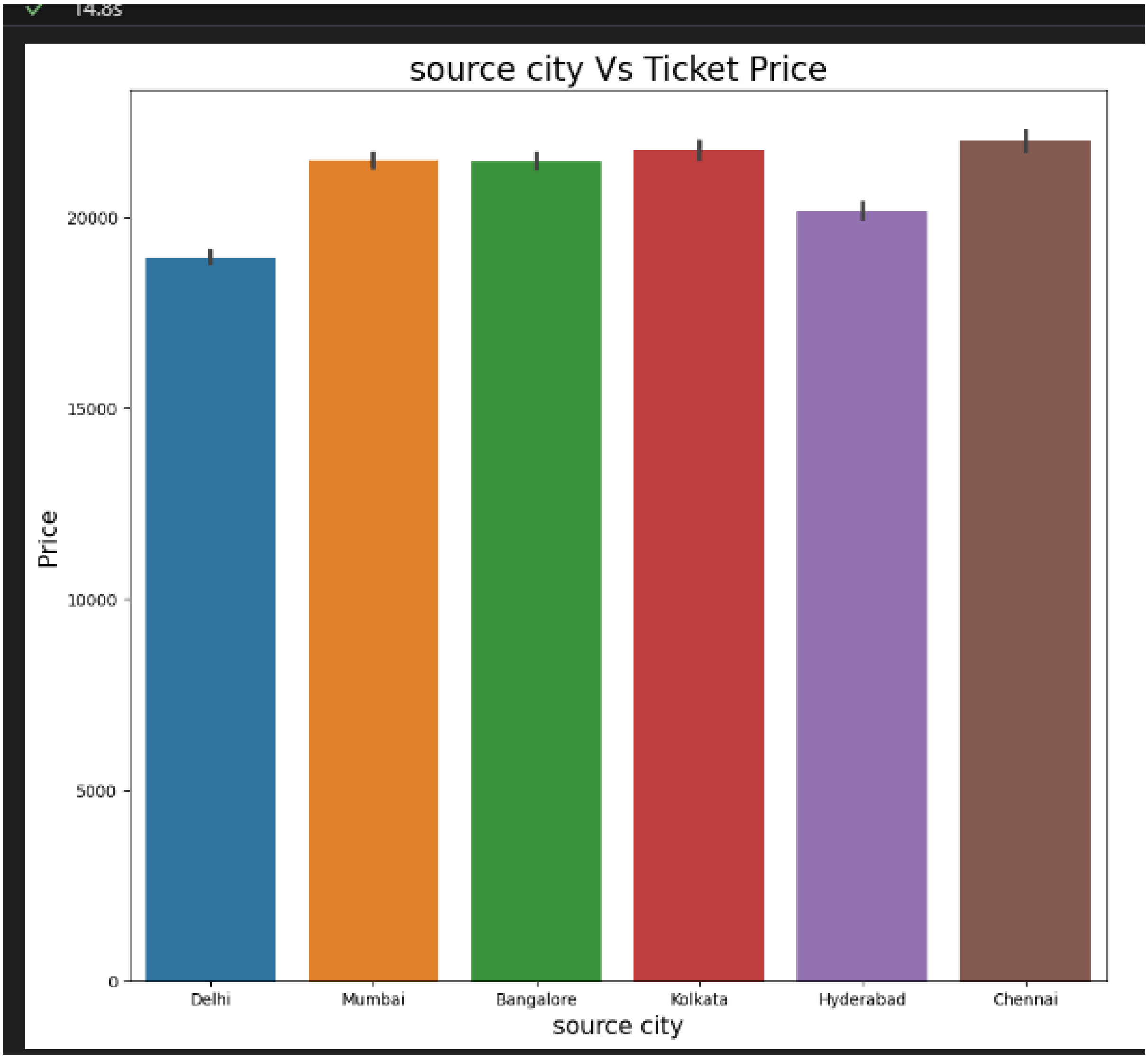
The ticket price is almost equal for the flights having the arrival; time at night and the morning



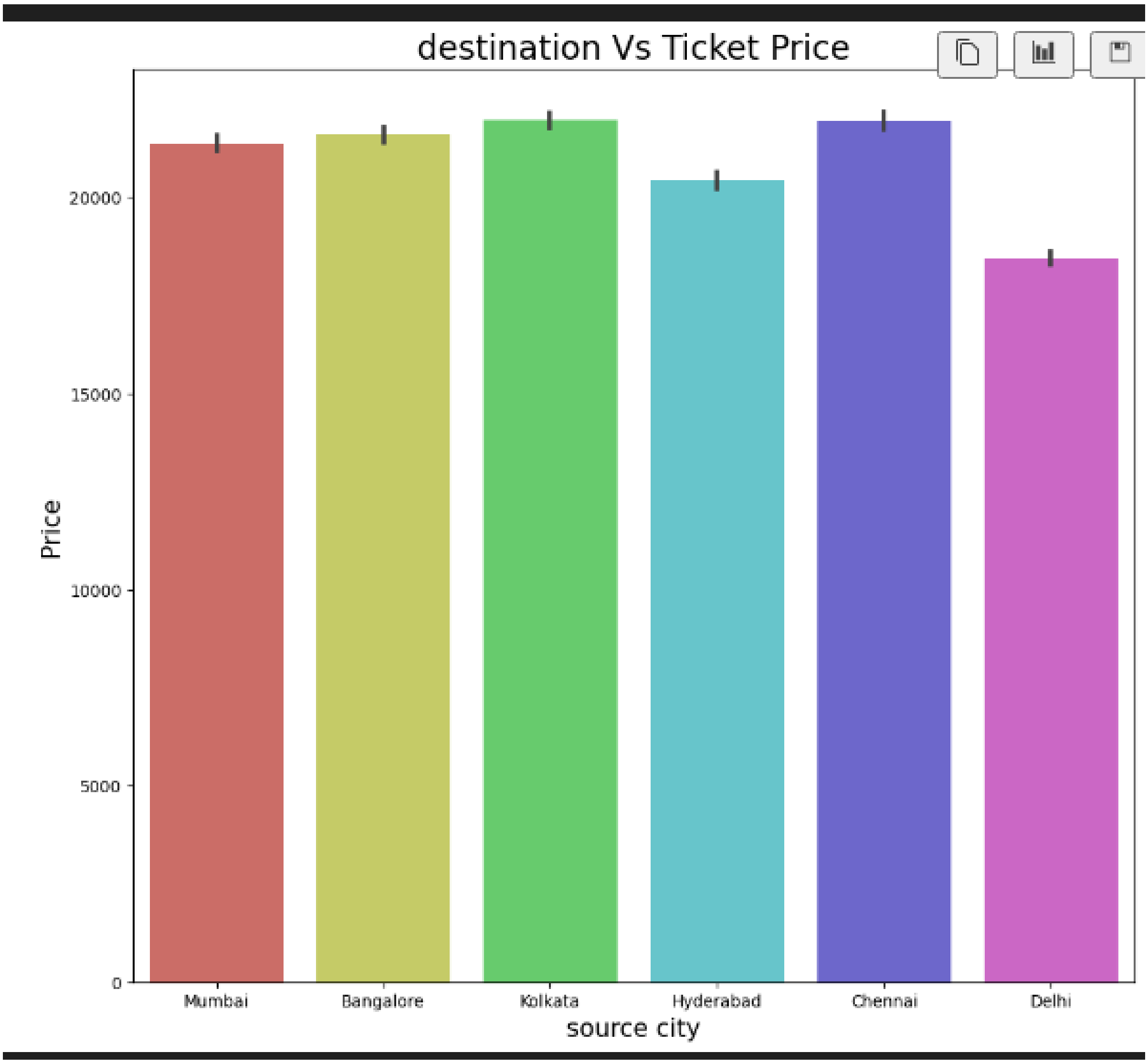
Bar plot for Source city vs ticket price and destination vs ticket price

From the box plot of the feature of source city vs the ticket price
We can infer that the flight ticket price is high for the city chennai and it is low for the city Delhi

We make the inference by seeing weather there is a significant change in the height of the bar plot



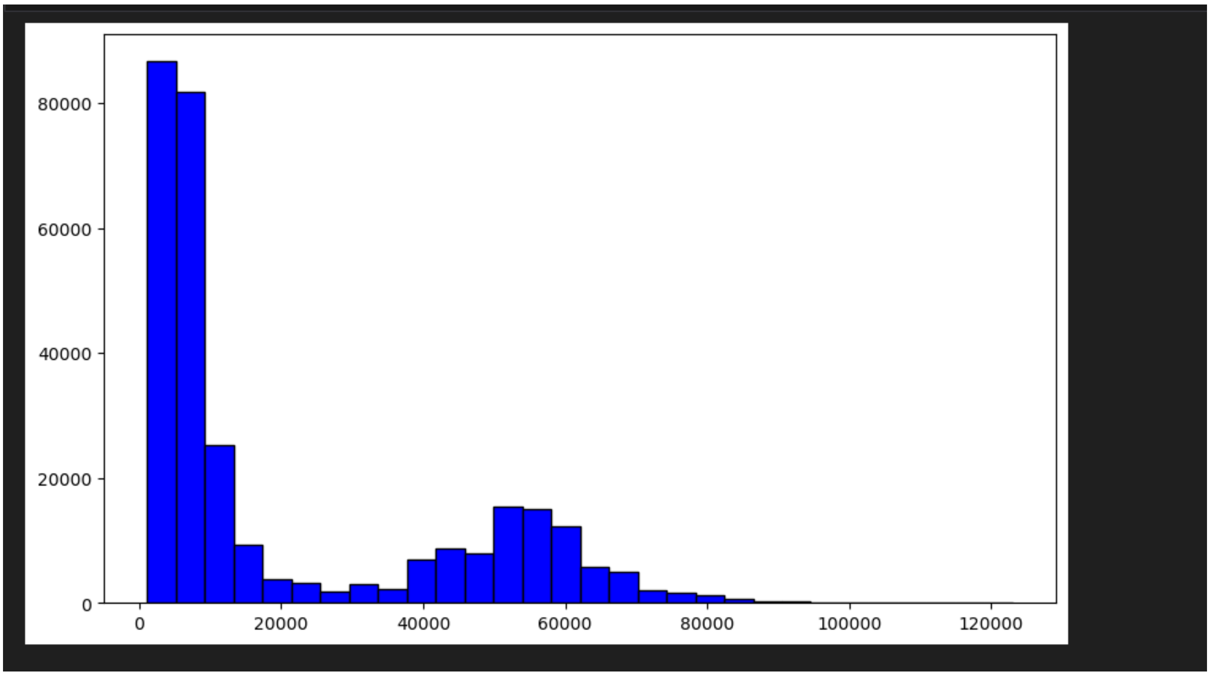
Next we plot the bar plot for the distribution of the destination vs the ticket price
 We can infer that the price is more when the destination city is chennai ans the price is low for the city delhi



Histogram

Next we plot the price distribution for the the flights
Analysing the price distribution the price distribution of the flight dataset using the histogram

we can get the central tendency ,spread and the shape of the distribution
The highest bar represents the central tendency of the of the price distribution
This gives us the most common price range



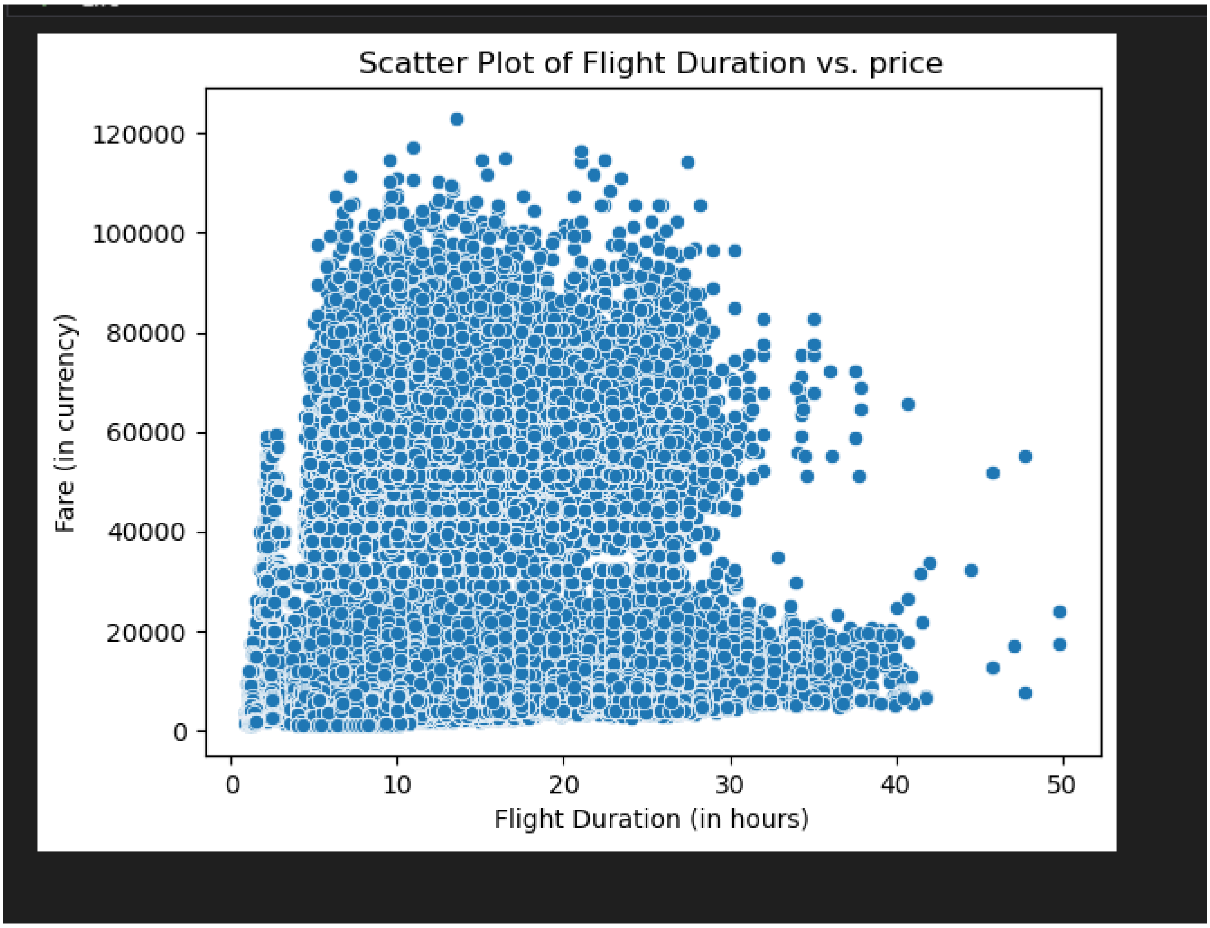
Scatter plot

If the scatter plot shows the upward trend it shows there is a positive correlation between the price and the duration of the flight

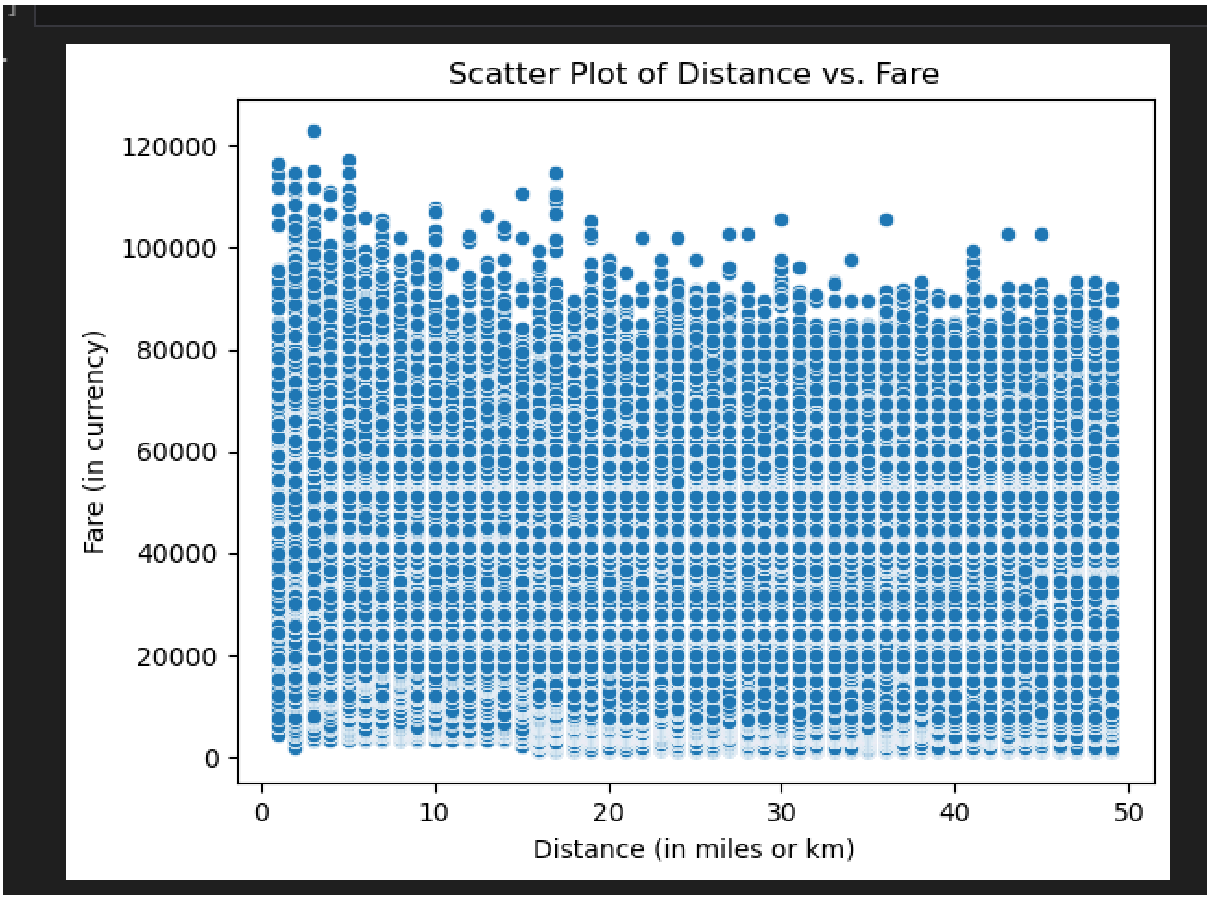
This shows that the longer flights tend to be more expensive

If there is no correlation if the points in the scatterplot are randomly distributed

There is no strong correlation between the price and the duration of the flight

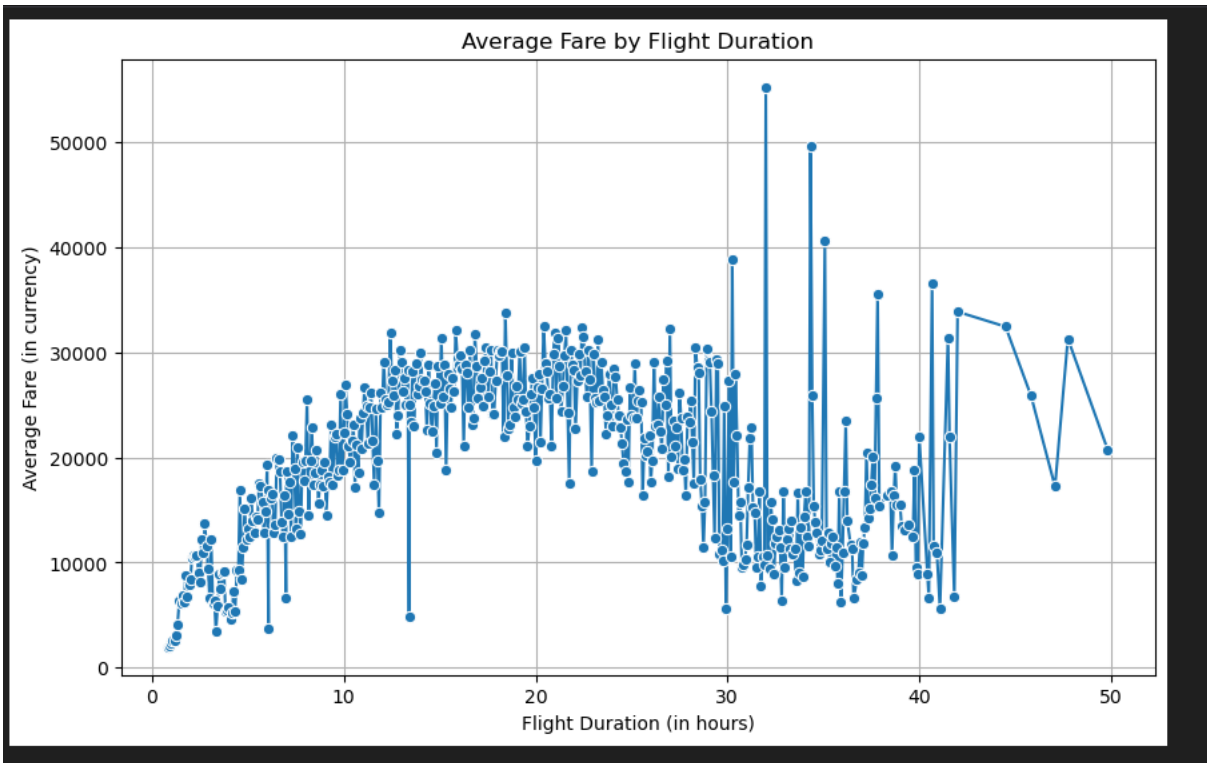


Then we did the scatter plot for the features between distance vs the fare



Scatter Plot of Distance vs. Fare
Fare (in currency)
Distance (in miles or km)
0
20000
40000
60000
80000
100000
120000
0
10
20
30
40
50

Line Plot



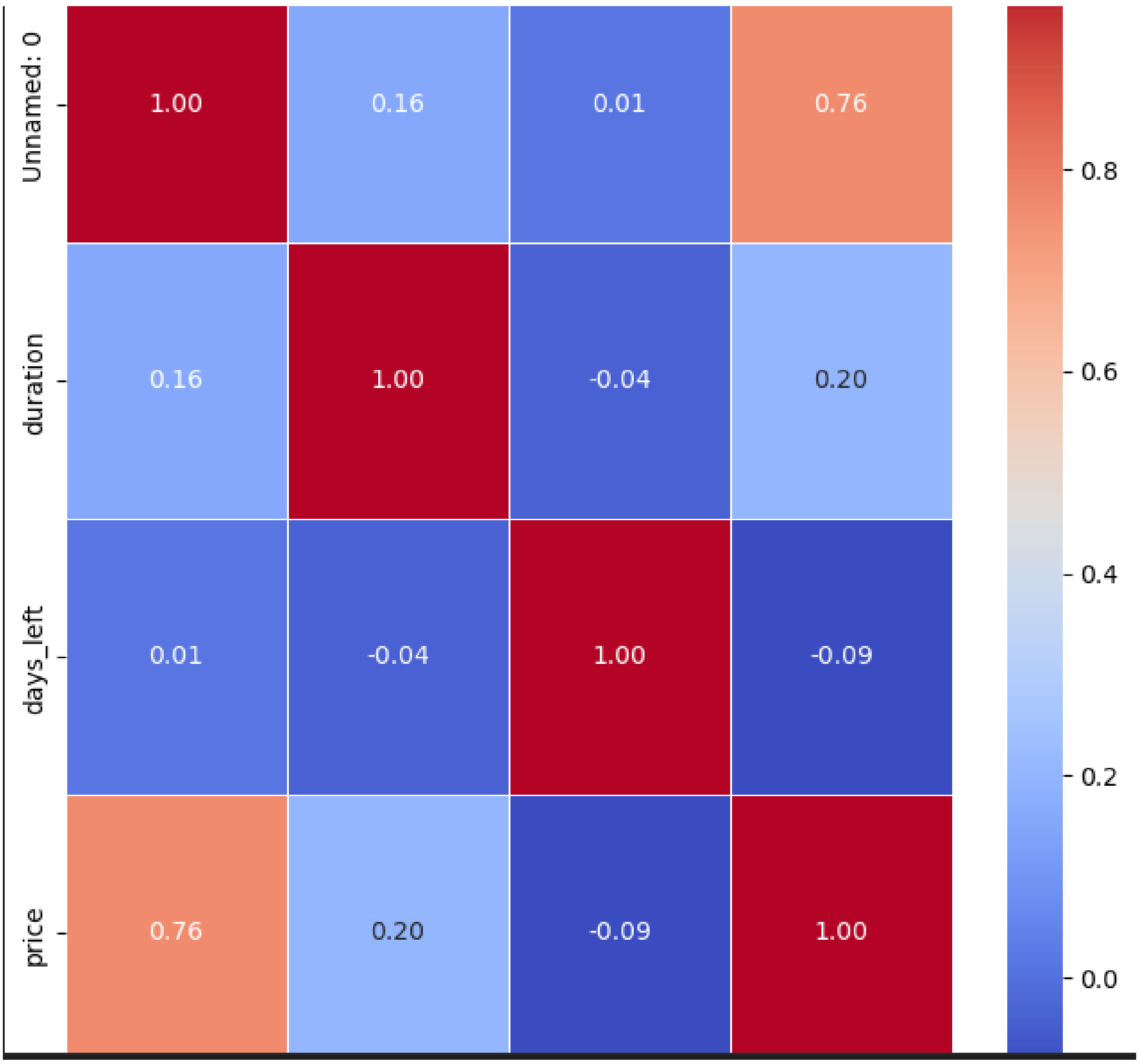
Average Fare by Flight Duration
Average Fare (in currency)
Flight Duration (in hours)
0
10000
20000
30000
40000
50000
0
10
20
30
40
50

Next we plot the line plot for the distribution between the average flight price and the flight duration from this we can infer that the

If he line tends to be up it indicates that on a the average the longer flights has the higher fare

If the line is relatively flat we can tell that the there may not be a strong relation between the flight duration and the average fare

Heatmap



Here the positive correlation is shown in one colour and the negative correlation is represented in another colour

The intensity of the colour represents the strength of the correlation
There is a strong positive correlation between the flight duration and the ticket price

Normalising the price and the duration

Before normalising the price feature was

```
df["price"]
✓ 0.0s

0          5953
1          5953
2          5956
3          5955
4          5955
          ...
300148    69265
300149    77105
300150    79099
300151    81585
300152    81585
```

The price feature after normalising using the

skew() ,log and the boxcox

The price feature normalised using boxcox

```
Price_boxcox = ((((df['price'])**price_lambda) - 1) / price_lambda)
print(Price_boxcox)
print(Price_boxcox.skew())
✓ 0.0s

0         3.561598
1         3.561598
2         3.561656
3         3.561637
4         3.561637
            ...
300148    3.773828
300149    3.780472
300150    3.782028
300151    3.783901
300152    3.783901
Name: price, Length: 300153, dtype: float64
0.1130734695419832
```

Normalising the duration

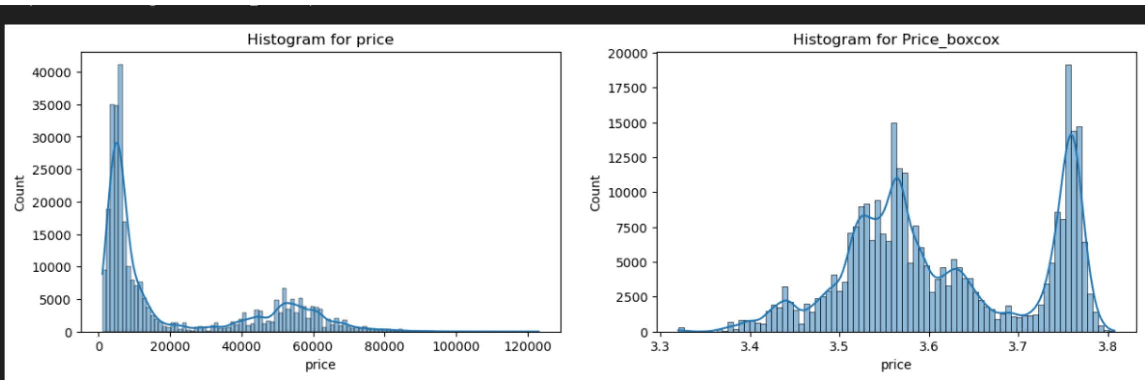
```
duration_boxcox, duration_lambda = sp.stats.boxcox(df['duration'])
print(duration_lambda)
✓ 3.1s

0.4829246270121536
```

```
duration_boxcox = ((((df['price'])**duration_lambda) - 1) / duration_lambda)
print(duration_boxcox)
print(duration_boxcox.skew())
✓ 0.1s

0         135.660875
1         135.660875
2         135.694390
3         135.683219
4         135.683219
             ...
300148    448.459565
300149    472.404093
300150    478.290623
300151    485.523143
300152    485.523143
Name: price, Length: 300153, dtype: float64
0.7812947768675217
```

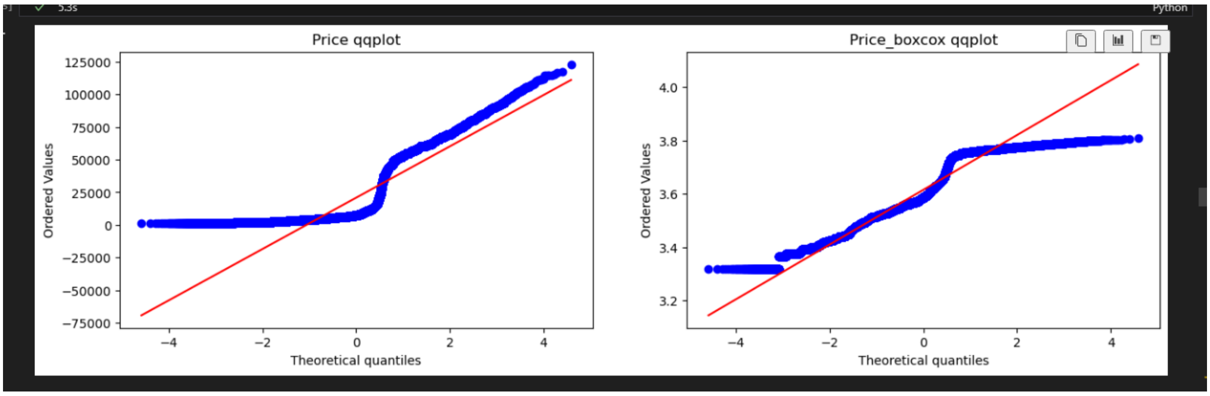
The next we plot the histogram for the the price feature before normalising and the after normalising



From these plots we can see that the before normalising the price feature is different and it is not having a bell shaped curve

The plot for the histogram_boxcox follows a bell shaped curve

Next we plot the price feature the qq plot before and after normalising



The points on the qq plots roughly follows the straight line it suggests that the normalised prices are approximately normalised

The deviations from the straight line indicates the departure or the deviation from the normality

The straight line tells us that the distribution of the price feature can follow normal distribution

Hypothesis testing

Here we are taking the the average price hypothesis

Null hypothesis

Here we can take the null hypothesis H0 as the:

The population mean of the flight fare is equal to 20889.660523133203

For doing the testing hypothesis here we are using a z test:

$$H_0 : \mu = 20889.660523133203 \text{ v/s } H_1 : \mu \neq 20889.660523133203$$

We performed a z test

The formula to do a z test is:

```
z_calc=(mean_price-5)/ ((var/2)/np.sqrt(n))
z_calc
```

The formula to get the p-value is

```
p_val=2*(1-sp.stats.norm.cdf(np.abs(z_calc)))
```

```
mean_price=df["price"].mean()
var=df['price'].var()
n=300153
alpha=0.05
z_calc=(mean_price-5)/ ((var/2)/np.sqrt(n))
z_calc
```
```
✓ 0.0s
0.04441835585446083
```

```
p_val=2*(1-sp.stats.norm.cdf(np.abs(z_calc)))
print("p-value",p_val)
if p_val<alpha:
    print("Reject the average price of the flight journey is not equal to 20889.660523133203")
else:
    print("accept that the averge price of the flight journey is 20889.660523133203")
```
```
✓ 0.0s
p-value 0.9645709302324248
accept that the averge price of the flight journey is 20889.660523133203
```

Here the z calc we calculates the z value and its value is 0.444183
And the p-value we got is :0.9645709302324248

Since the p-value is greater then the alpha which is 0.005 so we accept the null hypothesis and tells that the average price of the flight journey is 20889.660523133203